**In The United States Patent and Trademark Office
On Appeal From The Examiner To The Board
of Patent Appeals and Interferences**

In re Application of:	Brad K. Fayette
Serial No.	09/972,568
Filing Date:	October 5, 2001
Confirmation No.	5350
Group Art Unit:	2151
Examiner:	Kamal B. Divecha
Title:	Method and System for Communicating Among Heterogeneous Systems

Mail Stop - Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

Dear Sir:

Appeal Brief

Appellant has appealed to the Board of Patent Appeals and Interferences from the decision of the Examiner finally rejecting Claims 1-5, 11-17 and 20-22 as evidenced in the Final Office Action electronically sent February 9, 2007 and the Advisory Action mailed April 19, 2007. Appellant filed a Notice of Appeal on April 24, 2007. Appellant respectfully submits this Appeal Brief with the statutory fee of $500.00.

Table of Contents

Real Party In Interest

This application is currently owned by Fujitsu Limited as indicated by an assignment recorded on December 6, 2005, in the Assignment Records of the United States Patent and Trademark Office at Reel 017091, Frame 0587.

Related Appeals and Interferences

There are no known appeals or interferences which will directly affect or be directly affected by or have a bearing on the Board's decision regarding this appeal.

Status of Claims

Claims 1-5, 11-17 and 20-22 are pending in this application and all stand rejected under a final Office Action electronically sent February 9, 2007. Appellant presents Claims 1-5, 11-17 and 20-22 for appeal. Appendix A shows all pending claims.

Status of Amendments

All amendments presented by the Appellant were entered by the Examiner before the issuance of a Final Office Action.

Summary of Claimed Subject Matter

I. Concise Explanation of the Subject Matter Defined in Each Independent Claim

Claim 1 of the present application recites a method for processing a header portion of a message. The method includes establishing a legacy protocol that defines at least one legacy parameter for a header portion of a message and that defines a fixed legacy header length (as an example only and not by way of limitation, see ¶s 18-19; Figures 3 and 4, step 301). The method further includes receiving an inbound message having a header portion (as an example only and not by way of limitation, see ¶ 26; Figure 4, step 401) and allocating a memory portion from the computer memory, where the memory portion having a depth corresponding to said fixed legacy header length (as an example only and not by way of limitation, see ¶ 28; Figure 4, step 402). In addition, the method includes pushing the header portion of the inbound message onto the memory portion, thereby forming a received header (as an example only and not by way of limitation, see ¶s 27-28; Figure 4, step 403). The header portion is pushed onto the memory portion such that (i) the header portion is truncated to form the received header when a length of said header portion is greater than the depth of the memory portion corresponding to the fixed legacy header length and (ii) the header portion is not truncated when a length of the header portion is not greater than the depth of the memory portion (as an example only and not by way of limitation, see ¶s 30-31). Truncation causes any header parameters associated with an upgraded protocol to be removed from said header portion. Furthermore, the method includes processing said received header according to said legacy protocol (as an example only and not by way of limitation, see ¶ 28; Figure 4, step 404).

Claim 11 recites a method for processing a header portion of a message. Claim 11 includes all of the limitations of Claim 1 described above. Therefore, the same description and drawings references apply to these limitations. In addition, Claim 11 recites that the method also includes constructing a legacy header according to the legacy protocol (as an example only and not by way of limitation, see ¶s 21-22; Figure 3, step 303). Furthermore, Claim 11 recites appending said legacy header to outbound data to create an outbound message (as an example only and not by way of limitation, see ¶ 22; Figure 3, step 304) and sending the outbound message (as an example only and not by way of limitation, see ¶ 25; Figure 3, step 305).

Claim 15 of the present application recited software for processing at least one inbound message. The software is operable to establish a legacy protocol that defines at least one legacy parameter for a header portion of a message and that defines a fixed legacy header length (as an example only and not by way of limitation, see ¶s 18-19; Figures 3 and 4, step 301). The software is also operable to allocate a memory portion from the computer memory, the memory portion having a depth corresponding to the fixed legacy header length (as an example only and not by way of limitation, see ¶ 28; Figure 4, step 402). Furthermore, the software is operable to push the header portion of said inbound message onto the memory portion (as an example only and not by way of limitation, see ¶s 27-28; Figure 4, step 403). The data portion is pushed onto the stack first, and the header portion is pushed onto the memory portion such that (i) the header portion is truncated to form the received header when a length of the header portion is greater than the depth of the memory portion corresponding to the fixed legacy header length and (ii) the header portion is not truncated when a length of the header portion is not greater than the depth of the memory portion (as an example only and not by way of limitation, see ¶s 30-31). Truncation causes any header parameters associated with an upgraded protocol to be removed from the header portion. The software is further operable to process the received header according to the legacy protocol (as an example only and not by way of limitation, see ¶ 28; Figure 4, step 404).

Claim 20 of the present application recites software for processing inbound and outbound messages that is operable to establish a legacy protocol that defines at least one legacy parameter for a header portion of inbound and outbound messages and that defines a fixed legacy message length (as an example only and not by way of limitation, see ¶s 18-19; Figures 3 and 4, step 301). The software is also operable to receive an inbound message having a data portion (as an example only and not by way of limitation, see ¶ 26; Figure 4, step 401). Furthermore, the software is operable to allocate a memory stack from the computer memory, where the memory stack has a depth corresponding to the fixed legacy message length (as an example only and not by way of limitation, see ¶ 28; Figure 4, step

402). In addition, the software is operable to push the inbound message onto the memory stack thereby forming at least a portion of the inbound message (as an example only and not by way of limitation, see ¶s 27-28; Figure 4, step 403). The data portion is pushed onto the stack proximate to a bottom of the stack, and the header portion is pushed onto the memory portion such that (i) the header portion is truncated to form the received header when a length of the header portion is greater than the depth of the memory portion corresponding to the fixed legacy header length and (ii) the header portion is not truncated when a length of the header portion is not greater than the depth of the memory portion (as an example only and not by way of limitation, see ¶s 30-31). Truncation causes any header parameters associated with an upgraded protocol to be removed from the header portion. The software is also operable to process the portion of the inbound message according to the legacy protocol (as an example only and not by way of limitation, see ¶ 28; Figure 4, step 404). Moreover, the software is operable to construct a legacy header according to the legacy protocol (as an example only and not by way of limitation, see ¶s 21-22; Figure 3, step 303), append the legacy header to outbound data thereby creating an outbound message of the fixed legacy message length (as an example only and not by way of limitation, see ¶ 22; Figure 3, step 304), and send the outbound message (as an example only and not by way of limitation, see ¶ 25; Figure 3, step 305).

II. <u>Summary Description of Particular Embodiments</u>

Particular embodiments of the present invention may be better understood with reference to Figure 2, which illustrates a machine having an embodiment of the stateless protocol system 201 according to the invention. The machine can be a computer having a top layer application 202 which provides an interface for communicating over a network. The top layer application generally has a local data structure 211 for defining data according to a local format. Data manipulated by the top layer application 202 can be stored in a local buffer 212 memory device according the local format. The stateless protocol system 201 is provided in communication with the top layer application 202 and provides for conversion of data according to a protocol established by the stateless protocol system 201. The stateless protocol system 201 is provided with a legacy protocol 208 and can also include an upgraded protocol 209 which can be provided in addition to or in place of the legacy protocol 208.

The legacy protocol 208 and upgraded protocol 209 define formats for header information in the form of parameters which can provide communication information related to data being sent from one system to another, among other things. The legacy protocol 208 and upgraded protocol 209 can be stored in the memory or otherwise programmed into the stateless protocol system 201 as software or hard coded into logic circuits. (¶ 15)

Figure 3 of the present application illustrates an embodiment of the stateless protocol method. The method can be provided as an software or it can be hard coded into logic circuits of a system. The stateless protocol method establishes (step 301) a legacy protocol 208 by creating a format for a fixed length header as a combination of several parameters. The legacy protocol 208 that is established is initially provided to all instances of the stateless protocol method to be used by upper level machines in the network, such as the sending machine 110 and the receiving machine 140. In this manner, both the sending machine 110 and the receiving machine 140 are provided with a common understanding of the set of legacy parameters that make up a fixed length header according to the protocol. (¶ 18)

The legacy protocol can comprise one or more groups of parameters which when taken together comprise a fixed length header. Each parameter comprises a type and value pair, represented in this example as {type} and {value}, and when provided in combination can be used to define a fixed length header according to one embodiment of the invention. (¶ 19) The stateless protocol method can also establish an upgraded protocol (step 302) in an alternative embodiment. To establish an upgraded protocol, a set of parameters according to the legacy protocol are taken as a base of the header and then additional upgraded parameters for an upgraded version are appended to the legacy parameters. The additional parameters can be added to the end of the header away from the message according to the particular implementation of the protocol. (¶ 20)

The stateless protocol method implemented on the sending machine 110 constructs (step 303) a header intended for the receiving machine 140 based upon the fixed length header format defined by the legacy protocol 208. Alternatively, the header can be constructed according to the upgraded protocol 209. Once the header is created by the stateless protocol of the sending machine 110, the header is appended (step 304) to one end of the data to be sent. (¶ 22)

Figure 4 of the present application shows another embodiment of the stateless protocol method according to the invention. When a receiving machine 140 having a version of the stateless protocol method receives an inbound message (step 401), it can first strip the message of any additional header information associated with intermediate communication protocols of intermediate machines 120. It is believed that intermediate communication protocols between adjacent machines can provide header information including the length of the message being communicated. (¶ 26) In one embodiment of the stateless protocol method implemented on the receiving machine 140, each piece of data of the message packet is pushed (step 403) into in the memory 206 according to the order in which the data was received. Pushing data (step 403) on a memory 206 can have affect of flattening the data. As a result any additional header information, such as may be added by an upgraded version of the stateless protocol, is removed to prevent confusion when such a message is received by a machine having the legacy protocol 208. (¶ 27)

Since the sending machine 110 and receiving machine 140 share a common legacy version of the stateless protocol, the receiving machine 140 expects a message and header of a known length and can allocate a memory 206 of a corresponding depth for receiving the message or specifically for the header. If the receiving machine 140 expects to receive the same fixed format header as that which was sent by the sending machine 110, the length of the message and header received by receiving machine 140 matches the size of the memory 206 or stacks allocated by the stateless protocol implemented on the receiving machine 140. This result can be effected by only interpreting that portion of the header that is consistent with the protocol on the receiving machine 140. After the data has been pushed onto the memory 206, the stateless protocol system can interpret (step 404) the data according to the version of the protocol on that machine. (¶ 28)

The stateless protocol method permits a machine to upgrade its protocol from a legacy version of the stateless protocol to an upgraded version. In addition the stateless protocol method also permits further upgrades from previous upgraded protocols 209. In order to ensure that a receiving machine 140 can understand a header created by either the legacy protocol 208 or the upgraded protocol 209, the stateless protocol system and method may only upgrade a header in a consistent manner. For example, in establishing (step 302) an upgraded protocol 209, the parameters of the existing protocol are continued to be used in forming a header of a message and only additional data items are appended to the existing protocol. Furthermore, any additional parameters can be prepended to the legacy header 208 at the end of the header and away from the message. (¶ 29)

A receiving machine 140 having a legacy version of the stateless protocol can interpret a received message having an upgraded header. The receiving machine 140 expects to receive a message and header of a length set by the legacy format and allocates a memory or stack of corresponding depth. For example, when the receiving machine 140 receives a message, such as {3, 2, a, b, c, d, 1}, the message and header are added to the memory 206 first, and then are followed by the header. Since the memory 206 has been allocated only sufficient storage space for {a, b, c, d, 1} according to the legacy protocol 208, additional parameters of the upgraded header can be stripped off or ignored and the header becomes truncated or flattened. Thus, the parameters {3, 2} provided by the upgraded protocol 209 can be dropped because the memory has become occupied by the legacy protocol 208 which proceeded it. (¶ 30)

Thus, the pushing of data onto memory of a size determined by a legacy protocol 208 at a receiving machine 140 causes the message to be flattened by the memory 206 and thereby preserves only that information understood by the receiving machine 140 having the legacy format of the stateless protocol. As a consequence, the format of the header of sending machine 110 can be upgraded without disturbing the communications to other machines having an earlier version of the stateless protocol implemented in their systems. (¶ 31)

Ground of Rejection to be Reviewed on Appeal

Appellant requests that the Board review the following rejections:

- The Examiner's rejection of the specification and Claims 1-5, 11-17 and 20-22 under 35 U.S.C. § 112, first paragraph, as failing to comply with the written description requirement.

- The Examiner's rejection of Claims 1-2, 4-5, 11-16 and 20-22 under 35 U.S.C. § 103(a) as being unpatentable over *Denny* in view of *Birdwell.*

- The Examiner's rejection of Claims 3 and 17 under 35 U.S.C. § 103(a) as being unpatentable over *Denny* in view of *Birdwell* and further in view of *Taylor*.

Argument

The Examiner's rejections of Claims 1-5, 11-17 and 20-22 is improper, and the Board should withdraw these rejections for the reasons given below.

I. The Examiner's Rejection of Claims 1-5, 11-17 and 20-22 under Section 112 is Improper

Claims 1-5, 11-17 and 20-22 are rejected under 35 U.S.C. §112 , first paragraph, as failing to comply with the written description requirement. Specifically, the Examiner indicates that there is no support for the following language added to independent Claims 1, 11, 15 and 20 in Appellant's Response mailed on January 16, 2007: "wherein said header portion is not truncated when a length of said header portion is not greater than said depth of said memory portion."

Before this amendment, the claim language (for example, Claim 1) recited that the "header portion is truncated to form the received header if a length of said header portion is greater than said depth of said memory portion corresponding to said fixed legacy header length." The Examiner rejected this language as being indefinite because he argued that the "if" phrase "renders the claim indefinite because if the statement is false then the claim is irrelevant." *See* October 17, 2006 Office Action, pp. 2-3. Although Appellant believed that is was clear from this claim language that the header portion is not truncated if the length of the header portion is not greater than the depth of the memory portion, Appellant amended the claim to specifically state such in order to expedite allowance of this application.

However, in the Final Office sent February 9, 2007, the Examiner indicated that this added language is not supported and rejected the claims under section 112. Appellant respectfully disagrees. The present application supports the claim language regarding the truncation of the header in Figure 4 and the associated description (e.g., paragraphs 26-31). As an example, Appellant believes that it is clear from this description that the header is truncated if it is larger than the allocated memory space for the legacy protocol header (e.g., see paragraph 30). Also, Appellant submits that is easily understood from this description that the header would not be truncated if it fits within the allocated memory space (e.g., if a machine implementing a legacy protocol receives a message have a legacy header). *See, e.g.*, paragraph 23.

Finally, Appellant does not believe the Examiner's mention of the fact that the header may optionally be padded in association with steps 502 and 503 of Figure 5 is helpful to the Examiner's argument. In fact, this is an example of where the received header is not truncated when the length of the header portion is not greater than the depth of the memory portion. This figure is describing what happens when a header of a legacy protocol is received at a machine implementing an upgraded protocol which provides for a larger size header than the legacy protocol (thus the situation where the header does not fill the allocated memory space). On the other hand, as noted in the example of Figure 4, truncation is what may happen in particular embodiments when a header associated with the upgraded protocol is received at a machine implementing the legacy protocol (that allocates a smaller amount of space for the headers of received messages).

For at least the above reasons, Appellant respectfully submits that the claim amendments are fully supported by the specification. In the alternative, Appellant requests that the Board find that the added language to which the Examiner rejects is not needed to make the claims "relevant" (as the Examiner argued in the Office Action sent on October 17, 2006). Appellant respectfully requests allowance of these claims.

II. <u>The Examiner's Rejection of Claims 1-2, 4-5, 11-16 and 20-22 under Section 103 is Improper</u>

Claims 1-2, 4-5, 11-16 and 20-22 are rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent No. 5,544,325 issued to Denny ("*Denny*") in view of U.S. Patent No. 6,032,197 issued to Birdwell et al. ("*Birdwell*").

Claim 1 of the present application recites the following limitations:

> A method for processing a header portion of a message, comprising:
> establishing a legacy protocol, wherein said legacy protocol defines at

least one legacy parameter for a header portion of a message, and wherein said legacy protocol defines a fixed legacy header length;

receiving an inbound message having a header portion;

allocating a memory portion from the computer memory, said memory portion having a depth corresponding to said fixed legacy header length;

pushing said header portion of said inbound message onto said memory portion thereby forming a received header, wherein the header portion is pushed onto said memory portion such that said header portion is truncated to form the received header when a length of said header portion is greater than said depth of said memory portion corresponding to said fixed legacy header length and wherein said header portion is not truncated when a length of said header portion is not greater than said depth of said memory portion, such truncation causing any header parameters associated with an upgraded protocol to be removed from said header portion; and

processing said received header according to said legacy protocol.

Independent Claims 11, 15 and 20 recite similar, although not identical, limitations.

In order to establish a *prima facie* case of obviousness, three requirements must be met: (1) there must be some suggestion or motivation, either in the references themselves or in the knowledge available to one skilled in the art, to modify a reference or combine multiple references; (2) there must be a reasonable expectation of success; and (3) the prior art reference (or combination of references) must teach or suggest all of the claim limitations. M.P.E.P. § 2143. In the present case, a *prima facie* case of obviousness cannot be maintained at least because *Denny* and *Birdwell,* whether considered singly, in combination with one another, or in combination with information generally available to those of ordinary skill in the art at the time of the invention, fail to disclose all of the elements of the pending claims. Furthermore, there is no motivation to combine these references in the manner suggested by the Examiner.

For example, neither *Denny* or *Birdwell* disclose "wherein said legacy protocol defines a fixed legacy header length." The Examiner states that this limitation is disclosed in Figures 3 and 4 of *Denny*. However, *Denny* makes clear that the header (what it refers to as a "prefix") can be of variable length. *See Col. 5, lines 27-53 and Col. 6, lines 18-20.* Thus, Appellant believes that there is no disclosure of a *fixed* legacy header length. In the Response to Arguments section of the Final Office Action, the Examiner notes that *Denny* does disclose a fixed portion of the prefix; however, this is only a portion of the prefix and the total length of the prefix (fixed portion 42 and extended portion 44) is variable since the length of the extended portion is variable. As discussed below, the allocation of buffer space for the prefix is for the *entire* prefix, and not just for the fixed portion. Therefore, it is the entire prefix that is relevant to this analysis.

Furthermore, in the Response to Arguments section of the Final Office Action, the Examiner points to the background portion of *Denny* as evidence that *Denny* discloses a fixed length header. However, the background section is referring to the prior art and its associated problems (which *Denny* is attempting to solve). Specifically, *Denny* discloses that the prior art uses input message buffer capacities of particular systems that are not the same as the output buffer capacity of other systems. Thus, "a message having a prefix that is of the fixed specified length . . . will in many cases be incompatible for processing" by some systems. *Column 1, lines 47-53. Denny* addresses this incompatibility by <u>not</u> requiring a fixed prefix length. *E.g., see Column 2, lines 31-36 and 51-54.* Thus, although this background does describe a protocol using fixed prefix lengths, this is *not* a protocol that is used in the invention of *Denny* (which is what the Examiner refers to in rejecting the other limitations of this claim).

In addition, neither *Denny* or *Birdwell* disclose "allocating a memory portion from the computer memory, said memory portion having a depth corresponding to said fixed legacy header length." The Examiner has stated that this limitation is disclosed in Figure 4 (item 70) and also currently cites to Column 2, lines 25-60 and Column 3, lines 15-23. However, *Denny* specifically discloses that the memory used to store the message is expanded as necessary to accommodate the size of the *entire* prefix – both the fixed and variable portions. *See* Col. 7, lines 20-33. For example, *Denny* specifically states that "If the capacity of the input data buffer is insufficient to hold the prefixes 42, 44 of the message 40," the system allocates additional memory to store the message. *Column 7, lines 27-31.* Therefore, *Denny* does not allocate a memory portion corresponding to a *fixed* header length, instead it *dynamically* allocates a memory portion to accommodate a *variable* header length.

In the Response to Arguments section of the Final Office Action, the Examiner characterizes the teachings of *Denny* by stating that "the receiving means allocates memory portion from the computer memory having the depth *at least* equal to the size of the prefix, i.e., having a depth corresponding to fixed header length or size" (emphasis added). Therefore, it appears that the Examiner is interpreting "said memory portion having a depth corresponding to said fixed legacy header length" as meaning that the memory portion has a depth *at least* equal to the fixed legacy header length. If so, Appellant respectfully submits that this is an incorrect interpretation of the quoted claim language, both based on its plain meaning and based on the broadest reasonable interpretation of the claim in light of the specification.

Also, because *Denny* discloses dynamically allocating a memory portion to accommodate a *variable* header length (so that the *entire* header portion/prefix can be stored), *Denny* necessarily does not disclose that a portion of the message is truncated if it is greater that the "depth of the memory portion." Instead, *Denny* discloses that the memory is increased as necessary to accommodate the message. For this reason, there is simply not a motivation to combine a teaching from any reference that discloses truncating a message that is greater than a memory depth with the teachings of *Denny*. This is the opposite of what is done in *Denny*. In the Response to Arguments section the Examiner questions: "why can't compression/truncation be considered as an alternative in solving the compatibility problem as indicated above." Appellant is not arguing that it is not an alternative (in fact, that is what Appellant is saying when stating that it is the *opposite* of what is disclosed in *Denny*). However, that does not mean that it would be obvious to take portions of two opposite alternatives (expanding memory to fit a variable header size vs. truncating a variable header size to fit a fixed memory size) and meld them together, as the Examiner tries to do in combining *Denny* and *Birdwell*. The two alternatives work towards opposite goals and thus actually teach away from the combination of any sort of truncation with the teachings of *Denny*. So not only is there not a motivation to combine (since there is nothing in the references suggesting that header truncation could be used in the invention of *Denny)*, there is actually a motivation *not* to combine the references.

For at least the reasons provided above, Appellant believes that Claim 1 is in condition for allowance. In addition, independent Claims 11, 15 and 20 recite similar, although not identical, limitations to those discussed above. Thus, Appellant believes that these claims are also in condition for allowance. Therefore, Appellant respectfully requests allowance of Claims 1, 11, 15 and 20, as well as the claims that depend from these independent claims.

III. The Examiner's Rejection of Claims 3 and 17 under Section 103 is Improper

Claims 3 and 17 are rejected under 35 U.S.C. § 103(a) as being unpatentable over *Denny* in view of *Birdwell* and further in view of U.S. Patent No. 5,206,822 issued to Taylor et al. ("*Taylor*"). Claims 3 and 17 depend from independent Claims 1 and 15, respectively. As discussed above, Appellant believes Claims 1 and 15 are in condition for allowance. Therefore, at least because they depend from an allowable independent claim, Appellant respectfully requests reconsideration and allowance of Claims 3 and 17.

Conclusion

Appellant has demonstrated that the present invention, as claimed, is clearly distinguishable over the prior art cited by the Examiner. Therefore, Appellant respectfully requests the Board of Patent Appeals and Interferences to reverse the final rejection of the Examiner and instruct the Examiner to issue a notice of allowance of all claims.

Please charge a fee in amount of $500.00 to cover the filing fee for this Appeal Brief to Deposit Account No. 02-0384 of BAKER BOTTS L.L.P. The Commissioner is also authorized to charge any other fees or credit any overpayments to Deposit Account No. 02-0384 of BAKER BOTTS L.L.P.

Respectfully submitted,

BAKER BOTTS L.L.P.
Attorneys for Appellant

[illegible]

Brian W. Oaks
Reg. No. 44,981

Date: 6/8/07

Correspondence Address:
Customer Number 05073

Appendix A: Claims on Appeal

1. (Previously Presented) A method for processing a header portion of a message, comprising:

establishing a legacy protocol, wherein said legacy protocol defines at least one legacy parameter for a header portion of a message, and wherein said legacy protocol defines a fixed legacy header length;

receiving an inbound message having a header portion;

allocating a memory portion from the computer memory, said memory portion having a depth corresponding to said fixed legacy header length;

pushing said header portion of said inbound message onto said memory portion thereby forming a received header, wherein the header portion is pushed onto said memory portion such that said header portion is truncated to form the received header when a length of said header portion is greater than said depth of said memory portion corresponding to said fixed legacy header length and wherein said header portion is not truncated when a length of said header portion is not greater than said depth of said memory portion, such truncation causing any header parameters associated with an upgraded protocol to be removed from said header portion; and

processing said received header according to said legacy protocol.

2. (Previously Presented) The method according to claim 1, further comprising:

establishing said upgraded protocol, wherein said upgraded protocol includes said at least one legacy parameter of said legacy protocol, wherein said upgraded protocol defines at least one upgraded header parameter for said header portion, and wherein said upgraded protocol defines a fixed upgraded header length;

wherein said memory portion has a depth corresponding to said upgraded header length;

wherein said received header of said inbound message is processed according to said upgraded protocol if at least one upgraded header parameter is pushed on the memory portion; and

wherein said received header of said inbound message is processed according to said legacy protocol when no upgraded header parameters are pushed on the memory portion.

3. (Previously Presented) The method according to claim 2 further comprising padding said memory portion with default padding values when said header portion of said inbound message does not fill said memory portion.

4. (Previously Presented) The method according to claim 1, wherein said legacy parameter comprises a value-type pair.

5. (Previously Presented) The method according to claim 1, wherein said inbound message includes a data portion and wherein said header portion is pushed onto said memory portion after said data portion.

6. (Canceled)

7. (Canceled)

8. (Canceled)

9. (Canceled)

10. (Canceled)

11. (Previously Presented) A method for processing a header portion of a message, comprising:

establishing a legacy protocol, wherein said legacy protocol defines at least one legacy parameter for a header portion of a message, and wherein said legacy protocol defines a fixed legacy header length;

receiving an inbound message having a header portion;

allocating a memory portion from the computer memory,[1] said memory portion having a depth corresponding to said fixed legacy header length;

pushing said header portion of said inbound message onto said memory portion thereby forming a received header, wherein the header portion is pushed onto said memory portion such that said header portion is truncated to form the received header when a length of said header portion is greater than said depth of said memory portion corresponding to said fixed legacy header length and wherein said header portion is not truncated when a length of said header portion is not greater than said depth of said memory portion, such truncation causing any header parameters associated with an upgraded protocol to be removed from said header portion;

processing said received header according to said legacy protocol;

constructing a legacy header according to said legacy protocol;

appending said legacy header to outbound data thereby creating an outbound message; and

sending said outbound message.

[1] Applicants recognize the antecedent basis issue and will address following this appeal.

12. (Previously Presented) The method according to claim 11, further comprising:

establishing said upgraded protocol, wherein said upgraded protocol includes said at least one legacy parameter of said legacy protocol, wherein said upgraded protocol defines at least one upgraded header parameter, and wherein said upgraded protocol defines a fixed upgraded header length;

wherein said memory portion has a depth corresponding to said upgraded header length;

wherein said received header of said inbound message is processed according to said upgraded protocol if at least one upgraded header parameter is pushed on the memory stack;

wherein said received header of said inbound message is processed according to said legacy protocol if no upgraded header parameters are pushed on the memory stack;

constructing an upgraded header according to said upgraded protocol; and

appending said upgraded header to outbound data.

13. (Previously Presented) The method according to claim 12 further comprising pushing said legacy parameter onto said memory portion before said upgraded parameter is pushed onto said memory portion.

14. (Previously Presented) The method according to claim 11 further comprising:

receiving said inbound message from an upper layer application having a header portion in an upper layer format; and

sending said outbound message to a lower layer application.

15. (Previously Presented) Software for processing at least one inbound message, the software embodied on a computer-readable medium and operable, when executed by a computer, to:

establish a legacy protocol, wherein said legacy protocol defines at least one legacy parameter for a header portion of a message, and wherein said legacy protocol defines a fixed legacy header length;

allocate a memory portion from the computer memory, said memory portion having a depth corresponding to said fixed legacy header length;

push said header portion of said inbound message onto said memory portion wherein said data portion[2] is pushed onto said stack[3] first, wherein the header portion is pushed onto said memory portion such that said header portion is truncated to form the received header when a length of said header portion is greater than said depth of said memory portion corresponding to said fixed legacy header length and wherein said header portion is not truncated when a length of said header portion is not greater than said depth of said memory portion, such truncation causing any header parameters associated with an upgraded protocol to be removed from said header portion; and

process said received header according to said legacy protocol.

16. (Previously Presented) The software according to claim 15, further operable to:

establish said upgraded protocol, wherein said upgraded protocol includes said at least one legacy parameters of said legacy protocol, wherein said upgraded protocol defines at least one upgraded header parameter, and wherein said upgraded protocol defines a fixed upgraded header length;

wherein said memory portion has a depth corresponding to said upgraded header length;

wherein said portion of said message is processed according to said upgraded protocol if at least one upgraded header parameter is pushed onto said memory portion; and

wherein said portion of said message is processed according to said legacy protocol if

[2] Applicants recognize the antecedent basis issue and will address following this appeal.

[3] Applicants recognize the antecedent basis issue and will address following this appeal.

no upgraded header parameters are pushed onto said memory portion.

17. (Previously Presented) The software according to claim 16 further operable to pad said memory portion with default padding values when said message does not fill said memory portion.

18. (Canceled)

19. (Canceled)

20. (Previously Presented) Software for processing inbound and outbound messages, the software embodied on a computer-readable medium and operable, when executed by a computer, to:

establish a legacy protocol, wherein said legacy protocol defines at least one legacy parameter for a header portion of inbound and outbound messages, and wherein said legacy protocol defines a fixed legacy message length;

receive an inbound message having a data portion;

allocate a memory stack from said computer memory, said memory stack having a depth corresponding to said fixed legacy message length;

push said inbound message onto said memory stack thereby forming at least a portion of said inbound message, wherein said data portion is pushed onto said stack proximate to a bottom of said stack, also wherein the header portion is pushed onto said memory portion such that said header portion is truncated to form the received header when a length of said header portion is greater than said depth of said memory portion corresponding to said fixed legacy header length and wherein said header portion is not truncated when a length of said header portion is not greater than said depth of said memory portion, such truncation causing any header parameters associated with an upgraded protocol to be removed from said header portion;

process said portion of said inbound message according to said legacy protocol;

construct a legacy header according to said legacy protocol;

append said legacy header to outbound data thereby creating an outbound message of said fixed legacy message length; and

send said outbound message.

21. (Previously Presented) The software according to claim 20, further operable to:

establish said upgraded protocol, wherein said upgraded protocol includes said at least one legacy parameter of said legacy protocol, wherein said upgraded protocol defines at least one upgraded header parameter, and wherein said upgraded protocol defines a fixed upgraded message length;

wherein said memory stack has a depth corresponding to said upgraded message length;

wherein said portion of said inbound message is processed according to said upgraded protocol if at least one upgraded header parameter is pushed onto said memory stack;

wherein said portion of said inbound message is processed according to said legacy protocol if no upgraded header parameters are pushed onto said memory stack;

construct an upgraded header according to said upgraded protocol; and

append said upgraded header to outbound data, wherein said at least one legacy parameter is proximate to said outbound data, thereby creating an outbound message of said fixed upgraded message length.

22. (Previously Presented) The software according to claim 21 further operable to:

receive said inbound message from an upper layer application having a data portion in an upper layer format; and

send said outbound message to a lower layer application.

Appendix B: Evidence

NONE

Appendix C: Related Proceedings

NONE